# Async-HFL: Efficient and Robust Asynchronous Federated Learning in Hierarchical IoT Networks

**Xiaofan Yu**[1], Ludmila Cherkasova[2], Harsh Vardhan[1], Quanling Zhao[1], Emily Ekaireb[1], Xiyuan Zhang[1], Arya Mazumdar[1], Tajana Šimunić Rosing[1]

[1] University of California San Diego
[2] Arm Research

IoTDI 2023

System Energy Efficiency Lab
seelab.ucsd.edu

NSF

TILOS

see

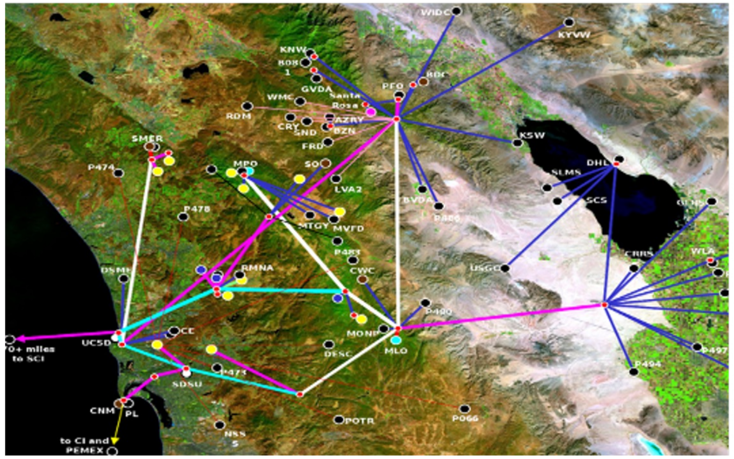
arm

UCSDCSE
Computer Science and Engineering

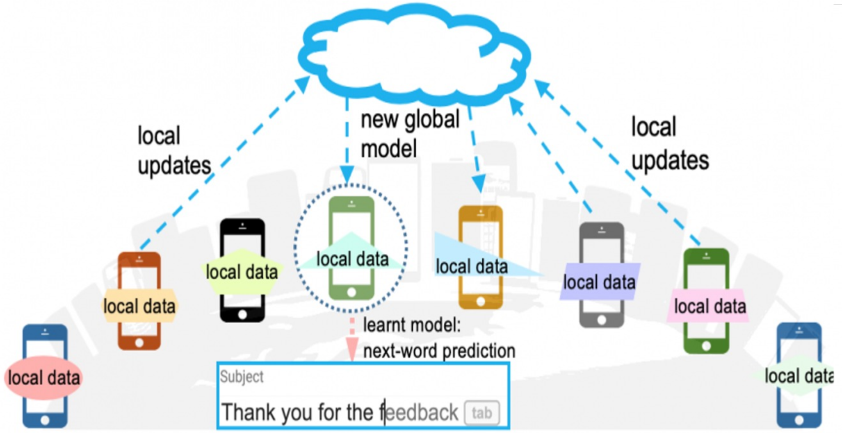
UC San Diego
HALICIOĞLU DATA SCIENCE INSTITUTE

# Federated Learning (FL)

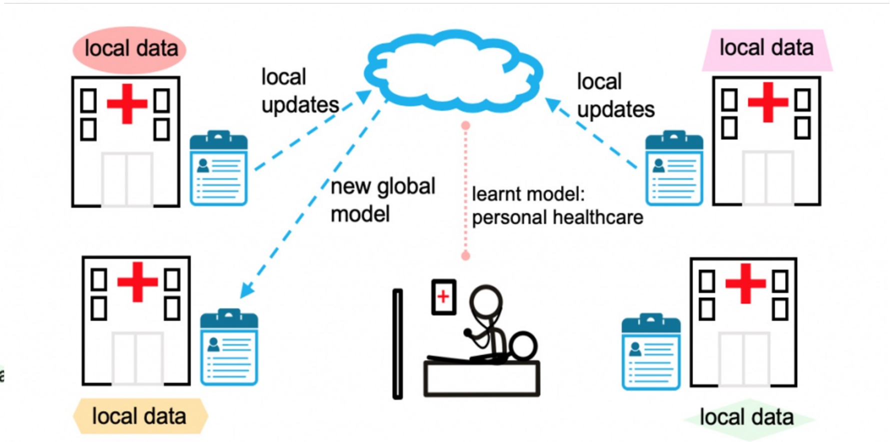
- Federated Learning is a machine learning technique that trains a model across multiple distributed edge devices without exchanging local data samples



Environmental Monitoring Sensor Networks



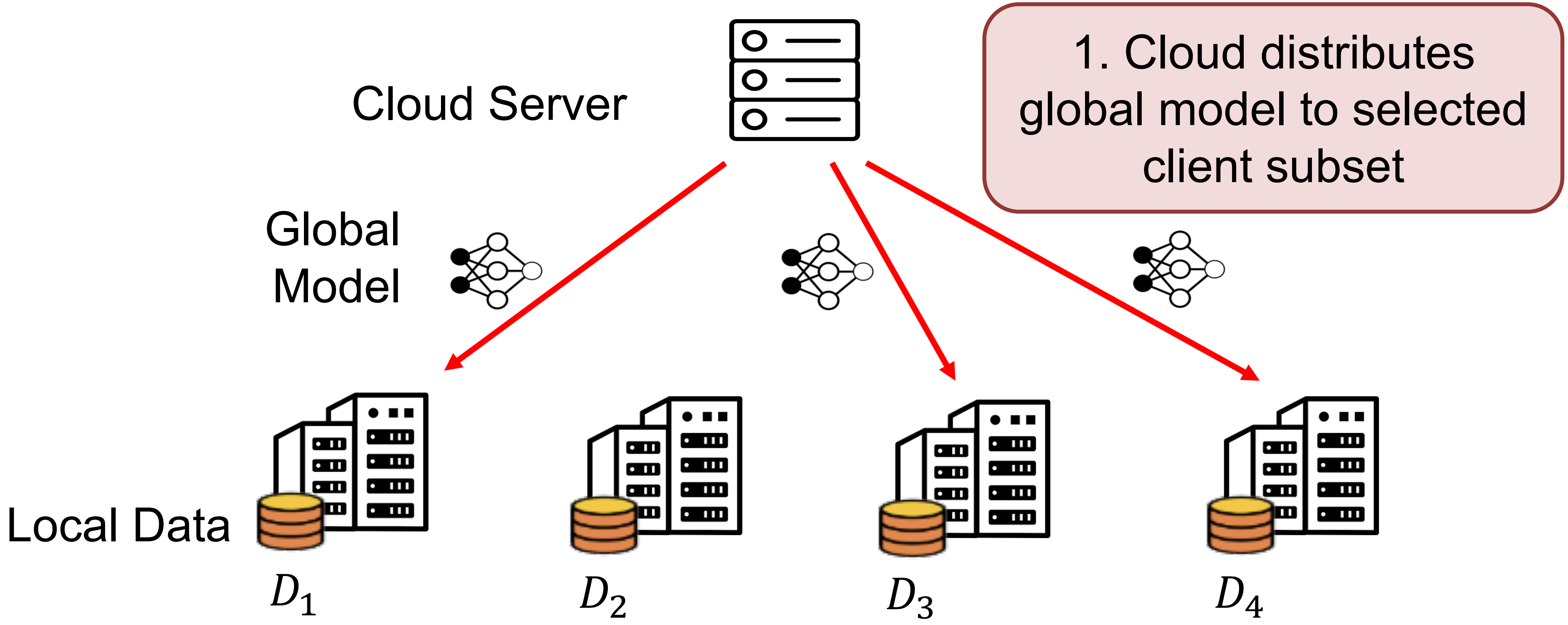

Next-word Prediction on Mobile Phones




Personal Healthcare Monitoring

Figure source: Li, Tian, et al. "Federated learning: Challenges, methods, and future directions." *IEEE Signal Processing Magazine* 37.3 (2020): 50-60.

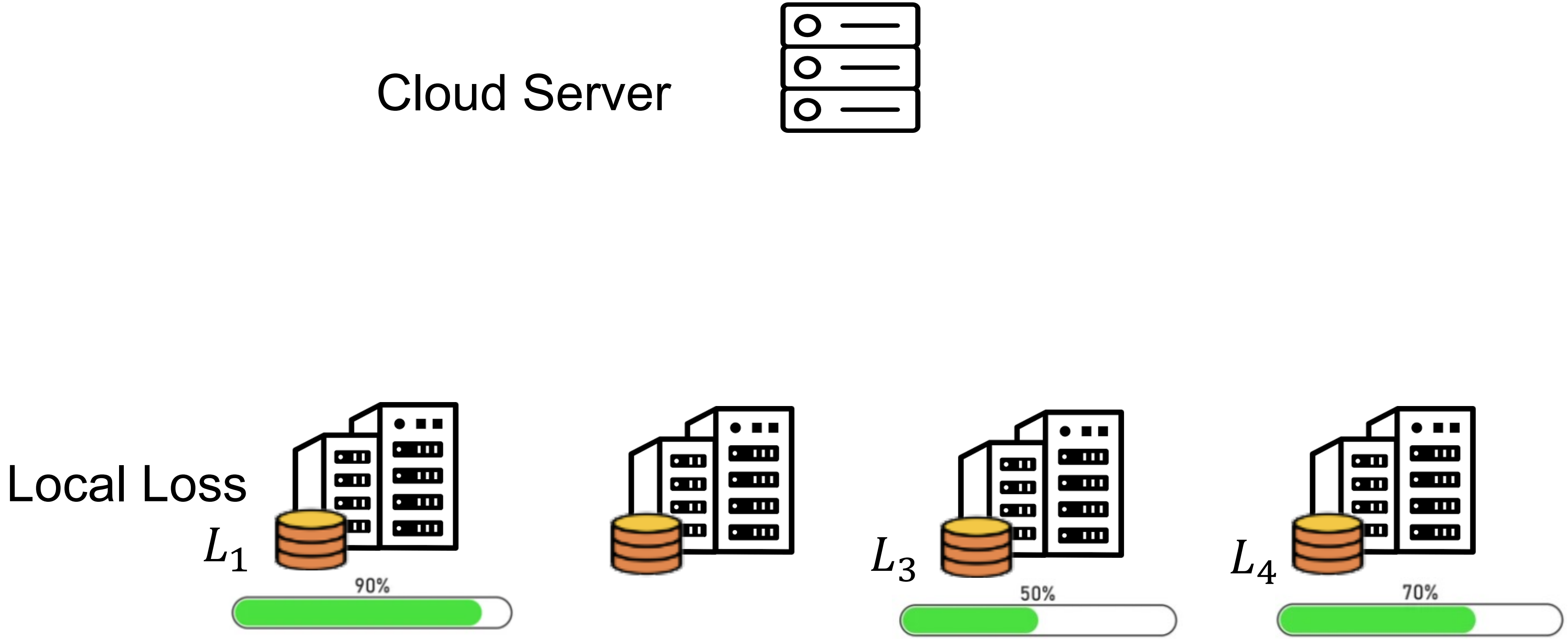
# Federated Averaging (FedAvg) [1]






[1] McMahan, Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." *Artificial Intelligence and Statistics*. PMLR, 2017.

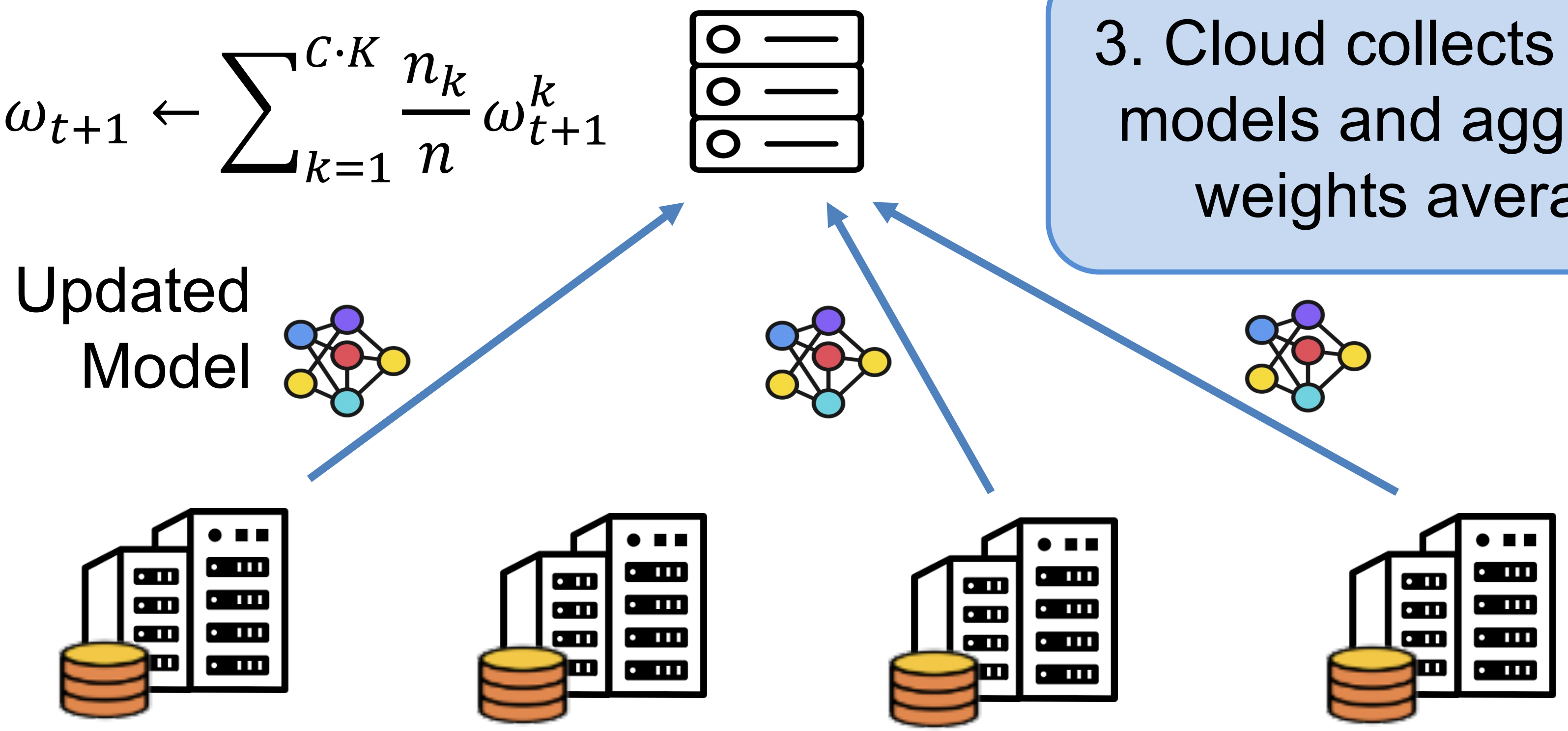
# Federated Averaging (FedAvg) [1]




2. Local training: SGD of $E$ epochs

[1] McMahan, Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." *Artificial Intelligence and Statistics*. PMLR, 2017.

# Federated Averaging (FedAvg) [1]




3. Cloud collects updated models and aggregates weights averagely

Sync Federated Learning (e.g., FedAvg)
may end up with significant slow down in IoT networks!!

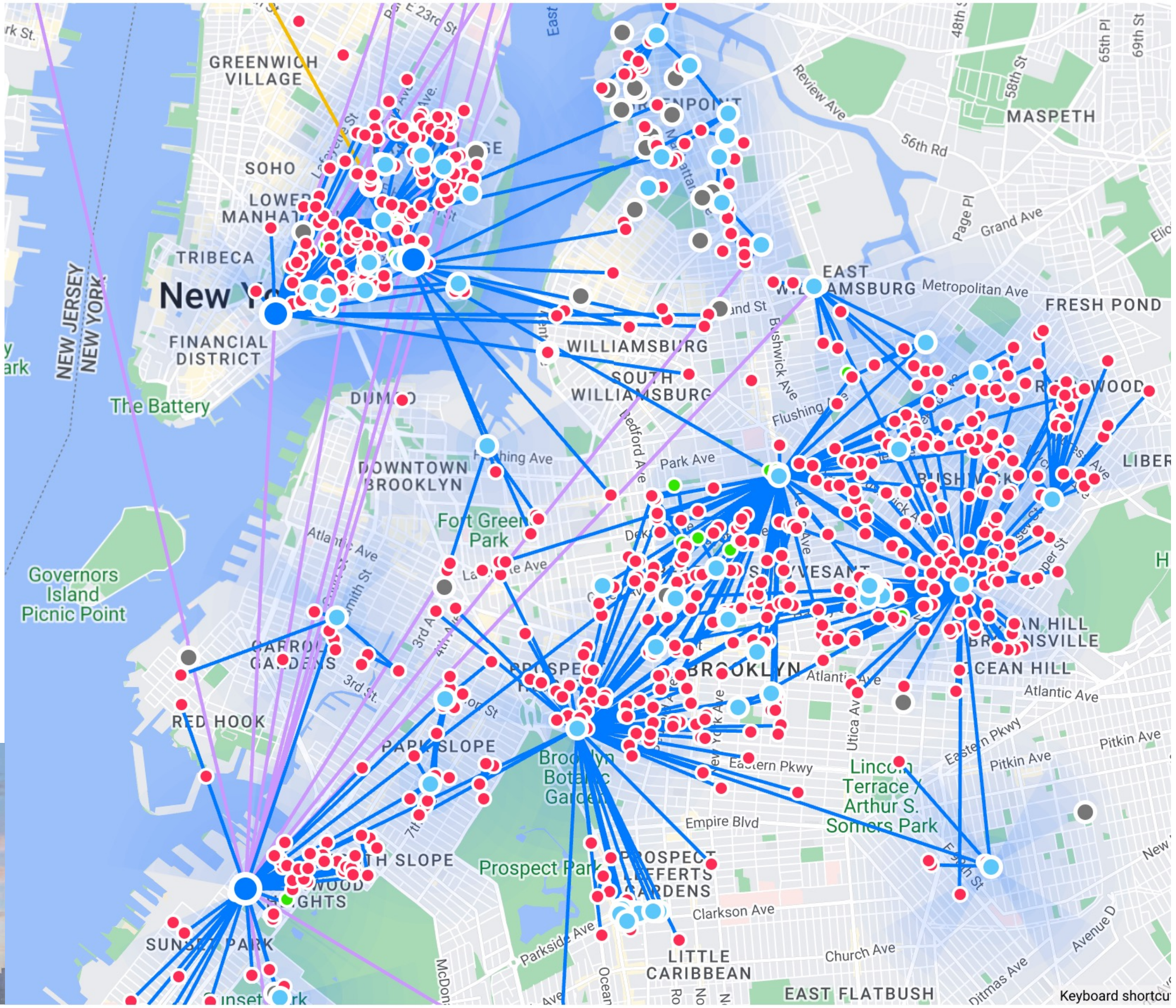


[1] McMahan, Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." *Artificial Intelligence and Statistics*. PMLR, 2017.

# Motivating Example: Federated Learning in NYCMesh

- NYCMesh [2] is a wireless mesh network in New York City, which mimics the future large-scale network backbone in smart cities

- Potential Federated Learning applications in NYCMesh:
  - Traffic monitoring
  - Noise monitoring
  - Video surveillance
  - …



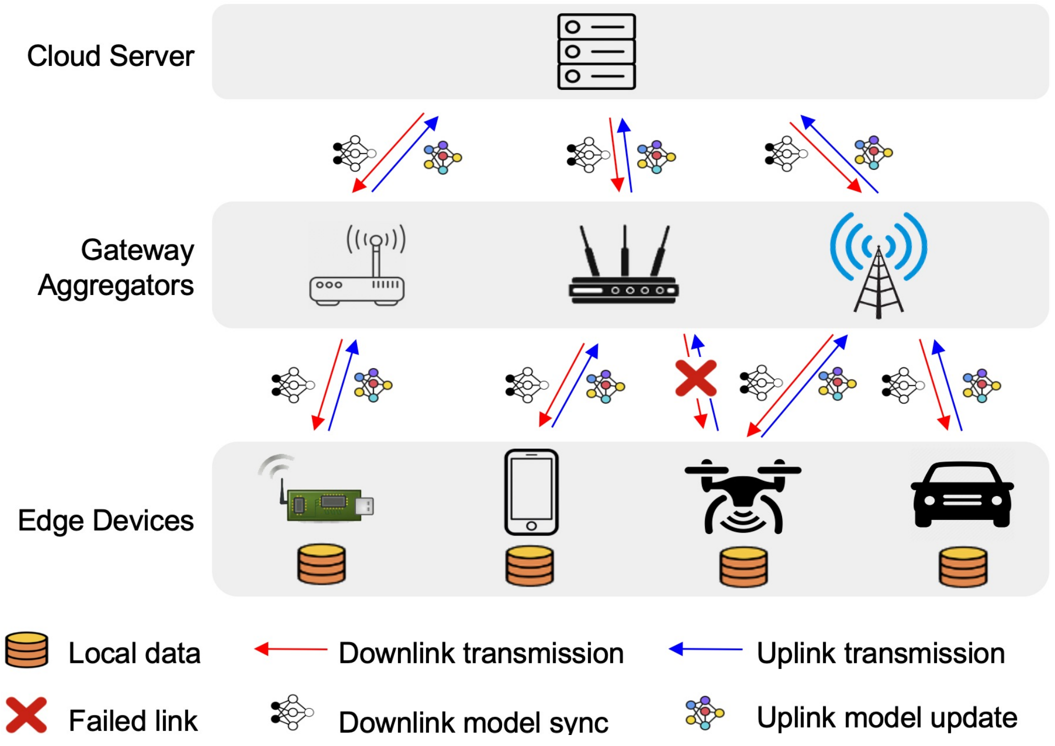

1316 Total | 60 AP | 1185 Non-hub | 65 Hub | 3 SN

Each hub has a camera


[2] NYCMesh. https://www.nycmesh.net/.

# Motivation: Unique Challenges of Hierarchical IoT Networks

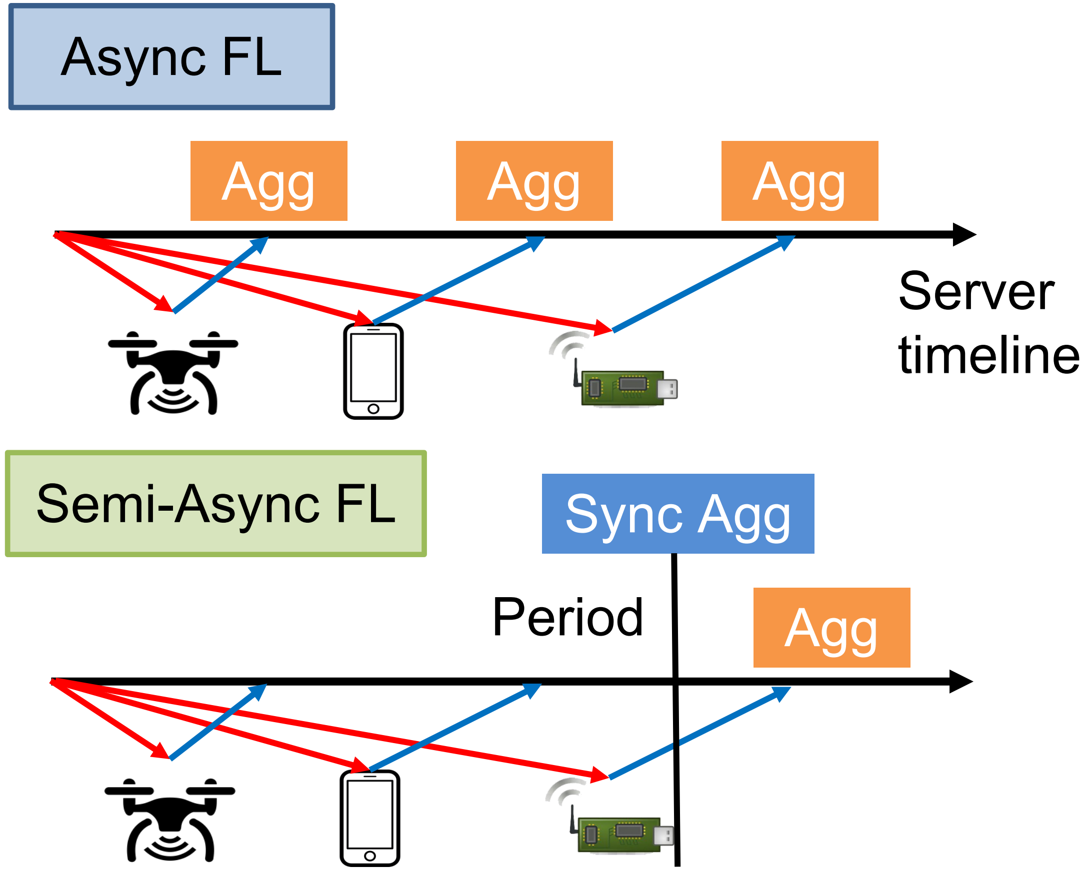
- Heterogeneous data distribution
- Hierarchical network organization (e.g., mesh networks)
- Heterogeneous system capabilities
  - Computation + Communication
- Unexpected stragglers (e.g., device or link failures)




Unique challenges of FL in Hierarchical IoT Networks!

# Previous Works

- Sync FL: based on FedAvg
  - Client selection: DivFL [ICLR'21], Oort [OSDI'21], PyramidFL [MobiCom'22]
  - (-) Significant delays in largely varied networks
- Async and semi-async FL
  - TrisaFed [IoT-J'22], FedBuff [AISTATS'22]
  - (-) Convergence challenges
- Hierarchical FL
  - Sync aggregation at gateway and cloud: SHARE [ICDCS'21]
  - Sync aggregation at gateway and async aggregation at cloud: RFL-HA [INFOCOM'21]
  - (-) Suffer from stragglers

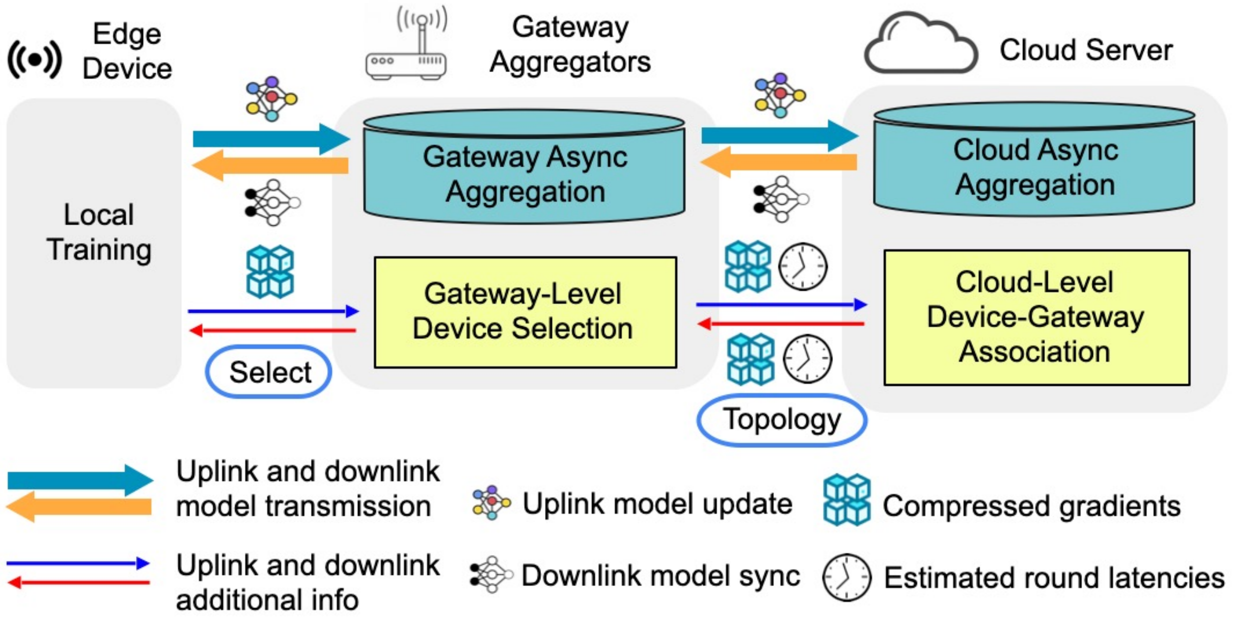



Async-HFL is the *first* end-to-end framework that addresses all challenges in a *hierarchical and unreliable* IoT networks!

# Our Contributions: Async-HFL

- Async-HFL is designed around three components to balance the data, system & network perspectives along with reacting timely to stragglers:

1. Async + hierarchical FL algorithm
2. Gateway-level device selection
3. Cloud-level device-gateway association




The managing framework of Async-HFL

# Theoretical Contribution: Convergence Analysis of Async-HFL

- **Asynchronous** aggregation at both the gateway and cloud inspired from *FedAsync* [3]
  - Two techniques are used to ensure convergence

**Async Aggregation**

Regularized local loss function at device

$$g^i(\omega^i;\omega_\tau) = L^i(\omega^i) + \frac{\rho}{2}\|\omega^i - \omega_\tau\|^2.$$

→ Penalize changes to global model thus ensure convergence

Staleness-aware aggregation at gateway/cloud

$$\alpha_h \leftarrow \alpha \cdot s(h-\tau)$$ Staleness function

$$\omega_h \leftarrow (1-\alpha_h)\omega_{h-1} + \alpha_h \omega_{new},$$

→ Aggregate based on the “age” of the model

- Convergence analysis:
  - Assume: $L$-smoothness, $\mu$-weak convexity, bounded gradients, bounded delay, sufficient regularization $\rho$

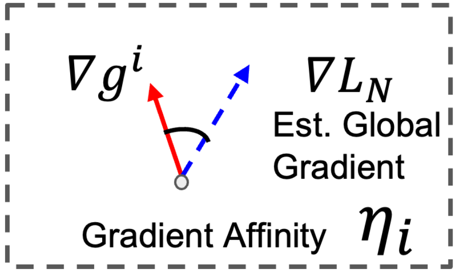


[3] Xie, Cong, Sanmi Koyejo, and Indranil Gupta. "Asynchronous federated optimization." *arXiv preprint arXiv:1903.03934* (2019).

# Modeling Data and System Heterogeneities

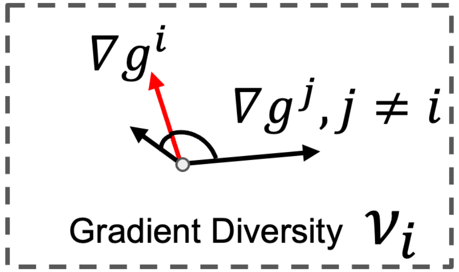
- **Data Heterogeneity:** we define a *learning utility* metric for each client based on the direction of compressed gradients

$$u_i = \eta_i + v_i$$




**Gradient Affinity:** Similarity between client's gradient and global gradient

**Gradient Diversity:** Dissimilarity between clients' gradients




- **System Heterogeneity:**
  - The computational and communication latencies on edge devices
  - The feasible sensor-gateway connections at time $t$ to account real-time link/device failures
  - Bandwidth limitation on sensor-gateway links

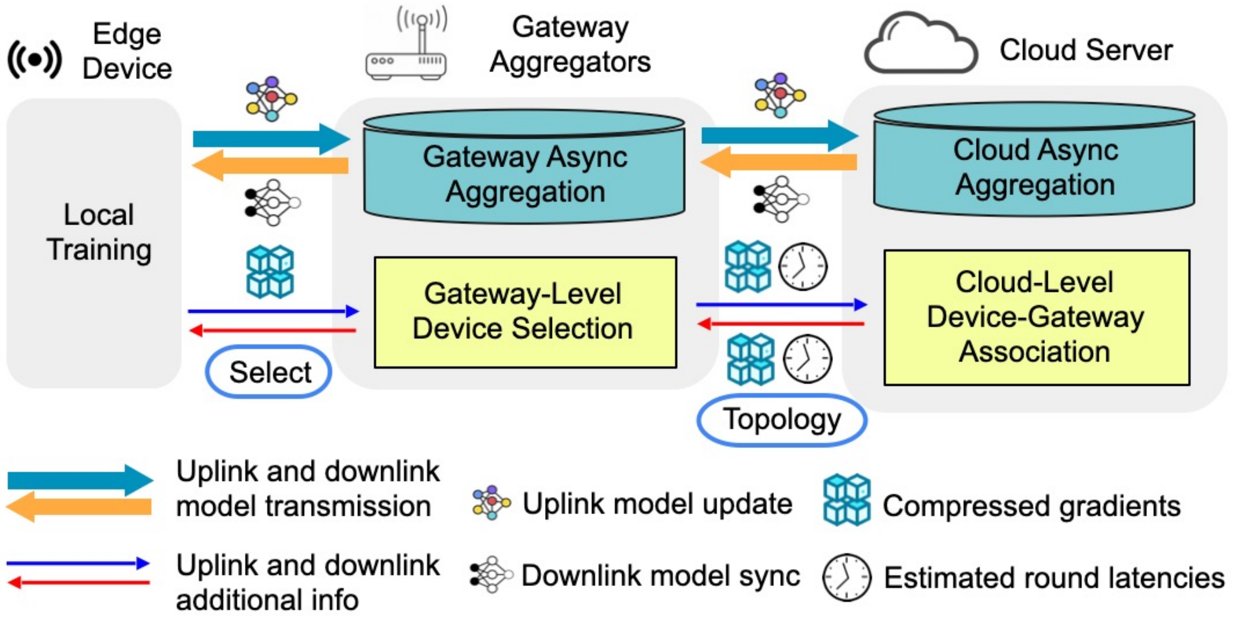
**Round latency:** $\tau_{ij} = \tau_{ji}^D + \tau_i^C + \tau_{ij}^U$

Downlink latency | Computational latency | Uplink latency

# Framework Management: Device Selection and Device-Gateway Association

- Gateway-level device selection and cloud-level device-gateway association collaboratively optimize practical convergence

    - Gateway-level device selection:
        - Real-time selection of devices to trigger local training
        - Balance *learning utility* and round latency
    - Cloud-level device-gateway association
        - Long-term network topology
        - Balance *learning utility* and throughput distribution under bandwidth limitation

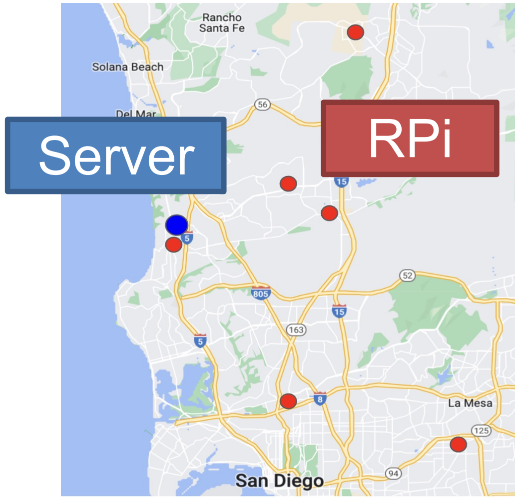


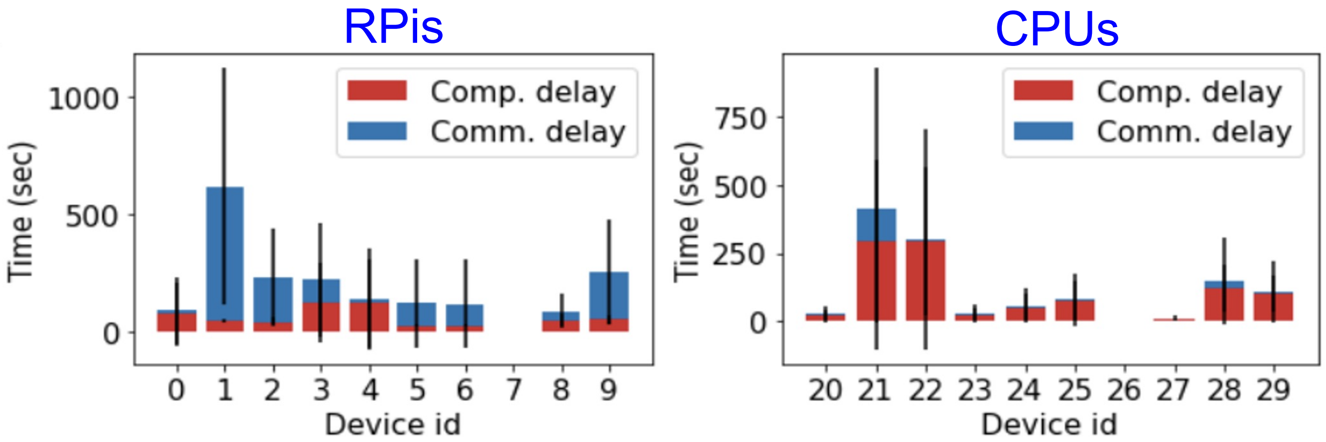

Both problems are formulated as Integer Linear Program and solved by the Gurobi solver.

# Experimental Setup

- We validate Async-HFL on a large-scale simulation and a physical deployment
  - **Large-scale simulation:** Simulation setup of NYCMesh in ns3-fl [4]
  - **Physical deployment:** 20 RPis and 20 CPUs
    - Implementation is based on FedML [5]

We spread the RPis in 7 houses, all connected to the home Wi-Fi



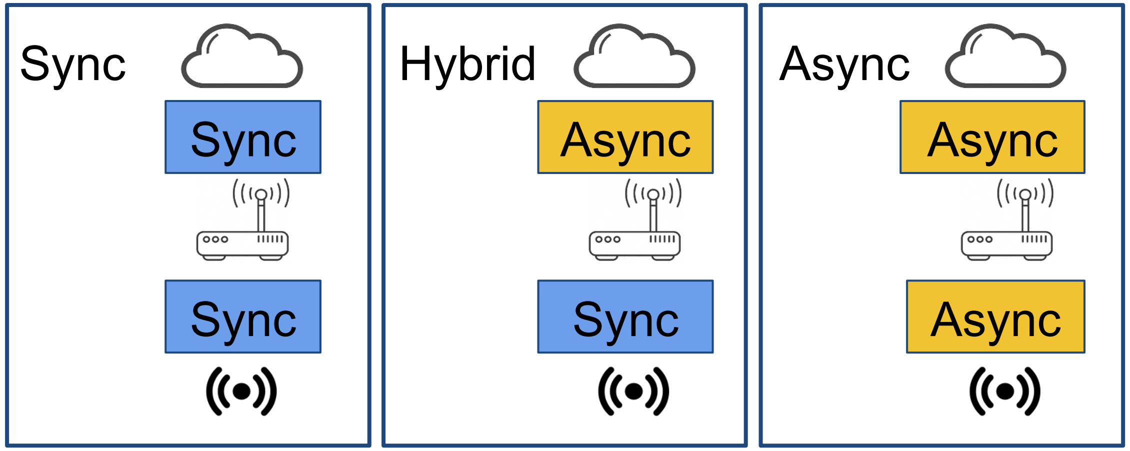




Round latency time break-ups on the RPis and CPUs


[4] Ekaireb, Emily, et al. “ns3-fl: Simulating Federated Learning with ns-3”, WNS3, 2022.
[5] He, Chaoyang, et al. "Fedml: A research library and benchmark for federated machine learning." *arXiv preprint arXiv:2007.13518* (2020).

# Experimental Setup (Cont.)

- **Baselines:**
  - **Sync:** random, TiFL [HPDC'20], DivFL [ICLR'21], Oort [OSDI'21]
  - **Hybrid:** RFL-HA [INFOCOM'21]
  - **Async:** random, high loss-first [SPAWC'21]




- **Metric:** The wall-clock convergence time to reach close-to-optimal accuracy

- **Datasets and models:**

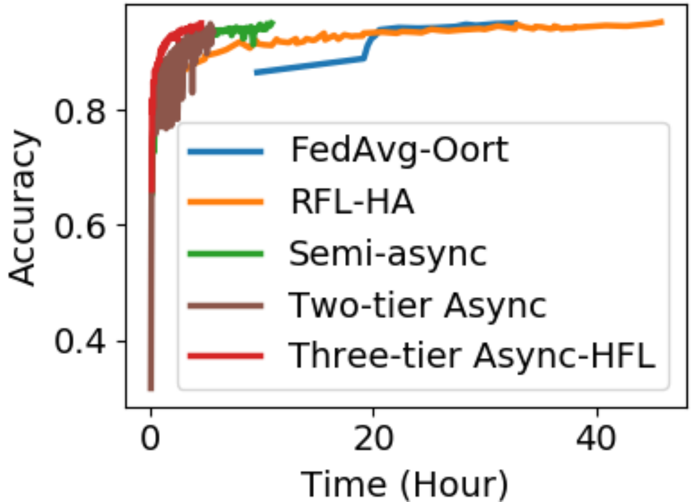
| Dataset | Models | Data Partition |
|---|---|---|
| MNIST, FashionMNIST | CNN | Synthetic |
| CIFAR-10 | ResNet-18 | Synthetic |
| Shakespeare, HPWREN | LSTM | Natural |
| HAR | MLP | Natural |

# Large-Scale Simulation Results

Convergence speedup of Async-HFL (over baselines)

| | Sync baselines | Semi-async baselines | RFL-HA | Async baselines |
|---|---|---|---|---|
| MNIST | 27.13x | 6.2x | 32.5x | 1.11x |
| FashionMNIST | 20.5x | 8.3x | 36.7x | 1.08x |
| CIFAR-10 | 44.3x | 2.3x | 12.3x | 1.09x |
| Shakespeare | 0.31x | 0.59x | 0.71x | 1.19x |
| HAR | 10.3x | 2.7x | 10.3x | 1.31x |
| HPWREN | 19.5x | 2.4x | 19.5x | 1.11x |

- Async-HFL converges **1.08-1.31x** faster in wall-clock time, and saves up to **21.6%** regarding total communication costs compared to state-of-the-art async FL algorithm (with client selection) on all datasets



# Physical Deployment Results

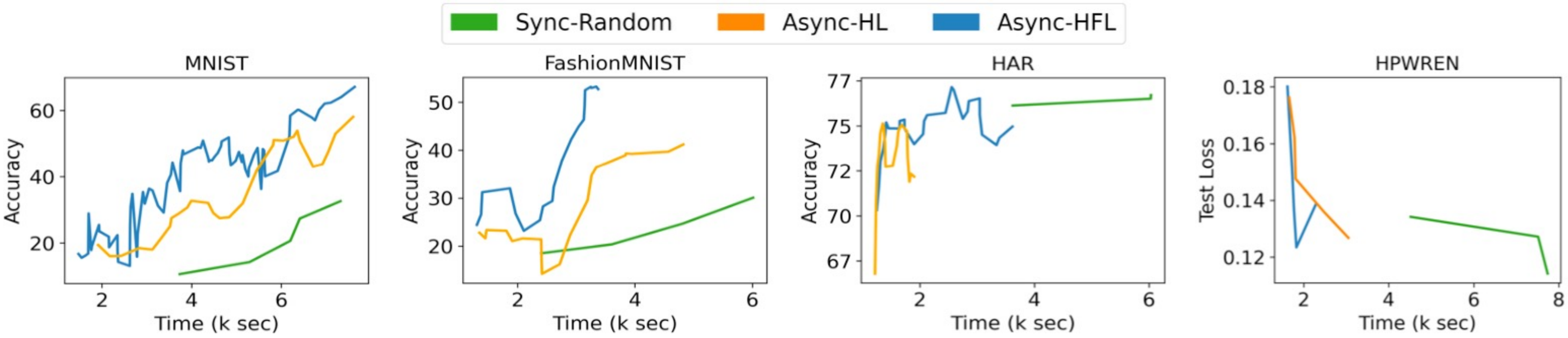



**Figure 8: Convergence results under wall-clock time on the physical deployment.**

- Async-HFL ends up with **higher accuracies on all datasets** than the state-of-the-art asynchronous baseline at similar time
- Our physical deployment presents largely heterogeneous round latencies and potential stragglers due to unexpected failures

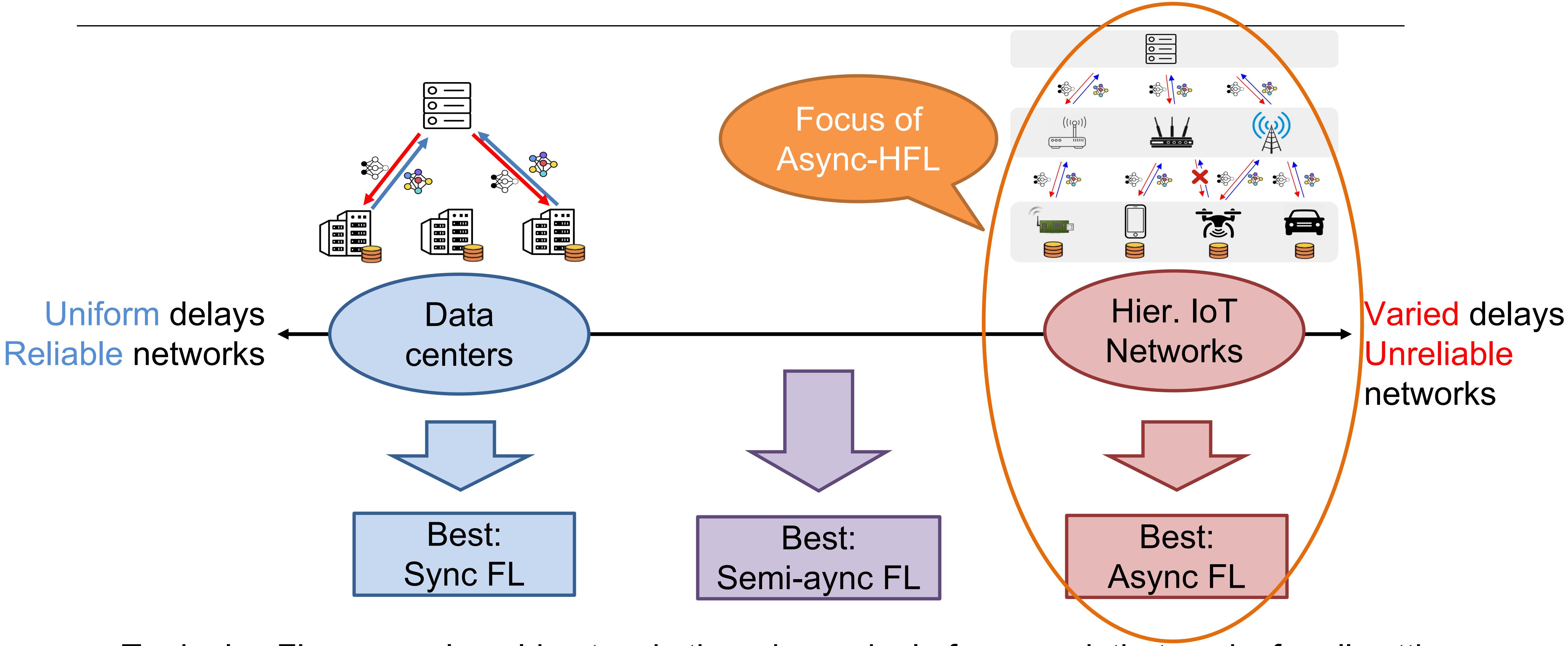
# Conclusion

- Existing FL designs suffer from significant delays and unexpected stragglers when considering hierarchical and unreliable IoT networks
- Async-HFL involves theoretical convergence analysis and practical framework design
  - Async-HFL conducts asynchronous aggregations at both the gateway and cloud
  - Async-HFL incorporates gateway-level device selection and cloud-level device-gateway association to enhance practical convergence
- Async-HFL converges 1.08-1.31x faster in wall-clock time, and saves saves up to 21.6% regarding total communication costs compared to state-of-the-art async FL algorithm (with client selection) on all datasets
- Code is available at https://github.com/Orienfish/Async-HFL

# References


- Chai, Zheng, et al. "Tifl: A tier-based federated learning system." *HPDC* 2020.
- Balakrishnan, Ravikumar, et al. "Diverse client selection for federated learning via submodular maximization." *ICLR* 2021.
- Lai, Fan, et al. "Oort: Efficient Federated Learning via Guided Participant Selection." *OSDI* 2021.
- Wang, Zhiyuan, et al. "Resource-efficient federated learning with hierarchical aggregation in edge computing." IEEE *INFOCOM 2021*
- Hu, Chung-Hsuan, Zheng Chen, and Erik G. Larsson. "Device Scheduling and Update Aggregation Policies for Asynchronous Federated Learning." IEEE *SPAWC* 2021.
- Deng, Yongheng, et al. "SHARE: Shaping data distribution at edge for communication-efficient hierarchical federated learning." IEEE *ICDCS 2021.*
- Nguyen, John, et al. "Federated learning with buffered asynchronous aggregation." *AISTATS PMLR, 2022.*
- You, Linlin, et al. "A triple-step asynchronous federated learning mechanism for client activation, interaction optimization, and aggregation enhancement." *IEEE Internet of Things Journal 2022.*
- Li, Chenning, et al. "PyramidFL: A fine-grained client selection framework for efficient federated learning." *MobiCom* 2022.

# Our Insights of Practical FL Deployments

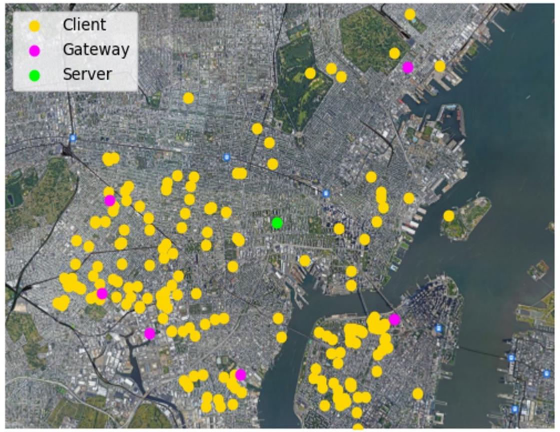



- To deploy FL on a real-world network, there is no single framework that works for all settings

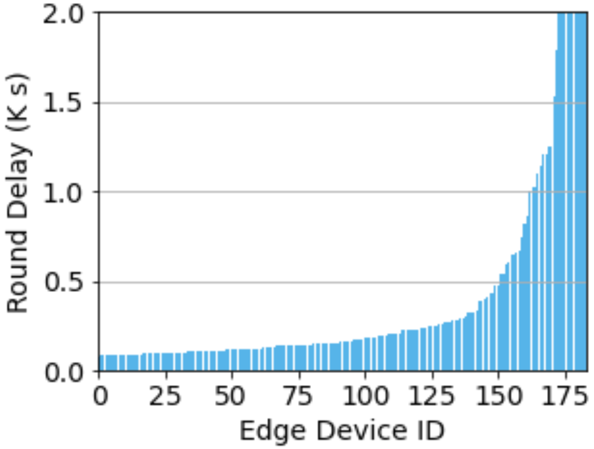
# Motivating Study: Federated Learning in NYCMesh

- We simulation FL in NYCMesh [2] using ns3-fl [3]
  - We extract the **latitude, longitude and rooftop height** of nodes, then feed the locations to the `HybridBuildingPropagationLossModel`
  - We add log-normal delay to simulate **long-tail latency distribution** in real deployments [4]

184 edge devices, 6 gateways, 1 server

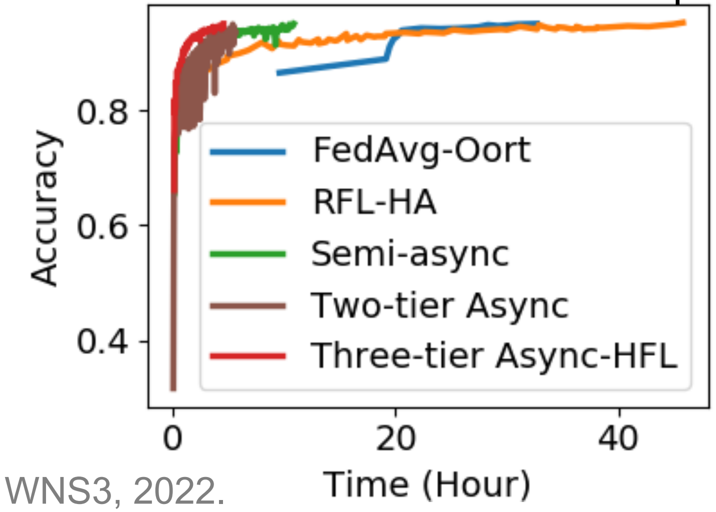


NYCMesh topology



Round latency distributions in the NYCMesh setup

- Major takeaways from the motivating study
  - **Async-HFL vs. sync, semi-async and RFL-HA baselines:** the three-tier Async-HFL achieves much faster convergence

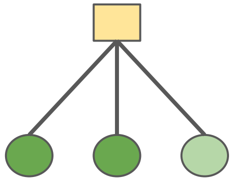




[2] NYCMesh. https://www.nycmesh.net/
[3] Ekaireb, Emily, et al. "ns3-fl: Simulating Federated Learning with ns-3", WNS3, 2022.
[4] Sui, Kaixin, et al. "Characterizing and Improving Wi-Fi Latency Large-Scale Operational Networks", MobiSys, 2016.

# Framework Management (Cont.): Device Selection and Device-Gateway Association

Gateway-Level Device Selection



**Input:** averaged round latency per link $\tau_{ij}$

latest compressed gradients per device $\nabla g^i$

**Intermediate:** learning utility $u_i$

**Output:** device for next gateway round $d_i$

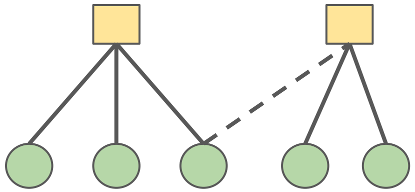
$$\text{(Device Selection at } j) \quad \max \sum_{\mathbf{I}_{t,ij}=1} d_i u_i \left(1/\tau_{ij}\right)^{\kappa} \tag{10a}$$

Bandwidth limitation

$$\text{s.t.} \quad d_i R_{ij} \le B_j, \quad \forall i \in \left\{ i \middle| \mathbf{I}_{t,ij} = 1 \right\} \tag{10b}$$

$$d_i \in \{0, 1\} \quad \forall i \in \left\{ i \middle| \mathbf{I}_{t,ij} = 1 \right\} \tag{10c}$$

Cloud-Level Device-Gateway Association



**Input:** $\tau_{ij}$, $\nabla g^i$, feasible connections $\boldsymbol{J}_t$

**Output:** device-gateway association $\boldsymbol{I}_t$

$$\text{(Association at cloud)} \quad \max u_{slack} - \phi R_{slack} \tag{11a}$$

Uniformly distributed learning utility and bandwidth

$$\sum_{i=1}^{N} \mathbf{I}_{t,ij}\, u_i \ge u_{slack}, \quad \forall j \in \mathcal{G} \tag{11b}$$

$$\sum_{i=1}^{N} \mathbf{I}_{t,ij}\, R_{ij}/B_j \le R_{slack}, \quad \forall j \in \mathcal{G} \tag{11c}$$

Feasible and valid connection

$$\mathbf{I}_{t,ij} \le \mathbf{J}_{t,ij}, \quad \forall i \in \mathcal{N}, j \in \mathcal{G} \tag{11d}$$

$$\sum_{j=1}^{G} \mathbf{I}_{t,ij} \le 1, \quad \forall i \in \mathcal{N} \tag{11e}$$

$$\mathbf{I}_{t,ij} \in \{0, 1\}, \quad \forall i \in \mathcal{N} \tag{11f}$$

Both problems are formulated as Integer Linear Program and solved by the Gurobi solver.

# Large-Scale Simulation Results

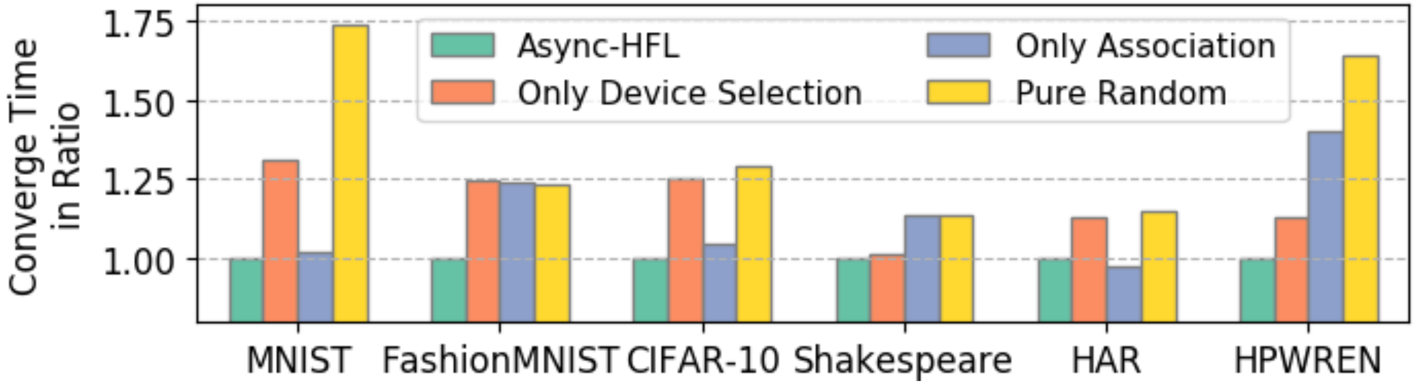
Table 6: Convergence speedup on large-scale simulations and various datasets. Bolded numbers reflect the best baseline result on each dataset.

| Dataset | Convergence time speedup of *Async-HFL* with respect to baselines | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Async-HL | Async-Random | Semi-async | RFL-HA | Sync-Oort | Sync-TiFL | Sync-DivFL | Sync-Random |
| MNIST | **1.11x** | 1.27x | 6.2x | 32.5x | 40.0x | 27.13x | 63.4x | 67.3x |
| FashionMNIST | **1.08x** | 1.49x | 8.3x | 36.7x | 20.5x | 32.8x | 73.4x | 96.8x |
| CIFAR-10 | **1.09x** | 1.40x | 2.3x | 12.3x | 44.3x | 59.0x | 62.0x | 61.7x |
| Shakespeare | 1.19x | 1.79x | 0.59x | 0.71x | **0.31x** | 2.39x | 5.87x | 5.46x |
| HAR | 1.31x | **1.22x** | 2.7x | 7.4x | 10.3x | 21.6x | 22.5x | 24.1x |
| HPWREN | **1.11x** | 1.48x | 2.4x | 12.8x | 19.5x | 26.5x | 27.7x | 31.4x |

- Async-HFL achieves significantly faster wall-clock time convergence than the sync and hybrid FL algorithms (with client selection) on most datasets
- Async-HFL converges 1.08-1.31x faster in wall-clock time, and saves saves up to 21.6% regarding total communication costs compared to state-of-the-art async FL algorithm (with client selection) on all datasets

# Ablation Studies

- *Question:* How does each of gateway-level device selection and cloud-level device-gateway association contribute to the convergence speedup separately?




- We evaluate (i) pure random selections, (ii) only device selection, (iii) only device-gateway-association, (iv) the full Async-HFL on all datasets
- Device selection dominates on MNIST and HAR, device-gateway association dominates on Shakespeare, while both modules contribute collaboratively on HPWREN